PATENTS

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicants: Ryuji Sato **Examin r:** Unassigned

Serial No.: To be assigned **Art Unit:** Unassigned

Filed: Herewith **Docket:** 14594

For: PROGRAM CREATION METHOD AND PROGRAM EXECUTION METHOD **Dated:** May 1, 2001

Assistant Commissioner for Patents
Washington, DC 20231

CLAIM OF PRIORITY

Sir:

Applicant in the above-identified application hereby claims the right of priority in connection with Title 35 U.S.C. §119 and in support thereof, herewith submits a certified copy of Japanese Patent Application No. 2000-136549 filed on May 10, 2000.

Respectfully submitted,

Paul J. Esatto, Jr.
Registration No. 30,749

SCULLY, SCOTT, MURPHY & PRESSER
400 Garden City Plaza
Garden City, New York 11530
(516) 742-4343

PJE:lf

CERTIFICATE OF MAILING BY "EXPRESS MAIL"

"Express Mail" Mailing Label Number: EL798805225
Date of Deposit: May 1, 2001

I hereby certify that this correspondence is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 C.F.R. §1.10 on the date indicated above and is addressed to the Assistant Commissioner of Patents and Trademarks, Washington, D.C. 20231.

Dated: May 1, 2001

Janet Grossman

#3

日　本　国　特　許　庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２０００年　５月１０日

出願番号
Application Number: 特願２０００－１３６５４９

出願人
Applicant (s): 日本電気株式会社

２００１年　３月　９日

特許庁長官
Commissioner,
Patent Office 及川耕造

出証番号　出証特２００１－３０１６６４３

【書類名】　特許願
【整理番号】　68501827
【提出日】　平成12年 5月10日
【あて先】　特許庁長官殿
【国際特許分類】　G06F 12/00
【発明者】
　【住所又は居所】　東京都港区芝五丁目７番１号　日本電気株式会社内
　【氏名】　佐藤　隆士
【特許出願人】
　【識別番号】　000004237
　【氏名又は名称】　日本電気株式会社
【代理人】
　【識別番号】　100088812
　【弁理士】
　【氏名又は名称】　▲柳▼川　信
【手数料の表示】
　【予納台帳番号】　030982
　【納付金額】　21,000円
【提出物件の目録】
　【物件名】　明細書　1
　【物件名】　図面　1
　【物件名】　要約書　1
【プルーフの要否】　要

【書類名】　　明細書

【発明の名称】　プログラム構築方法及びそのプログラムの実行方法

【特許請求の範囲】

【請求項１】　各々対応する処理コードを暗号化した暗号化コードをその実行時に相互に解除する第１及び第２のプログラムで構築したことを特徴とするプログラム構築方法。

【請求項２】　前記第１及び第２のプログラム各々は、前記暗号化コードと前記暗号化コードを解除する復号処理コードとを持つよう構築したことを特徴とする請求項１記載のプログラム構築方法。

【請求項３】　前記第１のプログラムは、全体の処理動作において奇数の順序で実行される暗号化コードを含み、

前記第２のプログラムは、前記全体の処理動作において偶数の順序で実行される暗号化コードを含むことを特徴とする請求項２記載のプログラム構築方法。

【請求項４】　前記第１及び第２のプログラム各々は、前記暗号化コードを解除する復号処理コードを持ち、

前記暗号化コードは、前記第１及び第２のプログラム各々の共有領域に含むよう構築したことを特徴とする請求項１記載のプログラム構築方法。

【請求項５】　前記復号処理コードは、前記処理コードに含むよう構築したことを特徴とする請求項１から請求項４のいずれか記載のプログラム構築方法。

【請求項６】　前記第１及び第２のプログラム各々において暗号解除を待つ側は、予め算出した算出時間だけ待機し、前記算出時間が経過すると前記暗号解除の有無にかかわらず次の処理を実行するよう構築したことを特徴とする請求項１から請求項５のいずれか記載のプログラム構築方法。

【請求項７】　前記第１及び第２のプログラム各々は、ソフトウェアデバッガによる動的な解析で前記暗号化コードの相互解除に遅れが生じた時に異常終了するよう構築したことを特徴とする請求項１から請求項６のいずれか記載のプログラム構築方法。

【請求項８】　前記暗号化コードは、前記第１及び第２のプログラムの両方

で復号しないと元の処理コードに戻らないよう構築したことを特徴とする請求項１から請求項７のいずれか記載のプログラム構築方法。

【請求項９】　前記第１及び第２のプログラムの起動時に前記第１及び第２のプログラムによって１番目の暗号化コードを復号してから前記第１のプログラムによってその復号された１番目の処理コードを実行し、前記第１及び第２のプログラムによって２番目の暗号化コードを復号してから前記第２のプログラムによってその復号された２番目の処理コードを実行するよう構築したことを特徴とする請求項１から請求項８のいずれか記載のプログラム構築方法。

【請求項１０】　第１及び第２のプログラムの実行時に前記第１及び第２のプログラム各々に対応する処理コードを暗号化した暗号化コードを相互に解除するようにしたことを特徴とするプログラム実行方法。

【請求項１１】　前記第１及び第２のプログラム各々は、前記暗号化コードと前記暗号化コードを解除する復号処理コードとを持つようにしたことを特徴とする請求項１０記載のプログラム実行方法。

【請求項１２】　前記第１のプログラムは、全体の処理動作において奇数の順序で実行される暗号化コードを含み、

前記第２のプログラムは、前記全体の処理動作において偶数の順序で実行される暗号化コードを含むようにしたことを特徴とする請求項１１記載のプログラム実行方法。

【請求項１３】　前記第１及び第２のプログラム各々は、前記暗号化コードを解除する復号処理コードを持ち、

前記暗号化コードは、前記第１及び第２のプログラム各々の共有領域に含むようにしたことを特徴とする請求項１０記載のプログラム実行方法。

【請求項１４】　前記復号処理コードは、前記処理コードに含むようにしたことを特徴とする請求項１０から請求項１３のいずれか記載のプログラム実行方法。

【請求項１５】　前記第１及び第２のプログラム各々において暗号解除を待つ側は、予め算出した算出時間だけ待機し、前記算出時間が経過すると前記暗号解除の有無にかかわらず次の処理を実行するよう構築したことを特徴とする請求

項１０から請求項１４のいずれか記載のプログラム実行方法。

【請求項１６】　前記第１及び第２のプログラムは、ソフトウェアデバッガによる動的な解析で前記暗号化コードの相互解除に遅れが生じた時に異常終了するようにしたことを特徴とする請求項１０から請求項１５のいずれか記載のプログラム実行方法。

【請求項１７】　前記暗号化コードは、前記第１及び第２のプログラムの両方で復号しないと元の処理コードに戻らないようにしたことを特徴とする請求項１０から請求項１６のいずれか記載のプログラム実行方法。

【請求項１８】　前記第１及び第２のプログラムの起動時に前記第１及び第２のプログラムによって１番目の暗号化コードを復号してから前記第１のプログラムによってその復号された１番目の処理コードを実行し、前記第１及び第２のプログラムによって２番目の暗号化コードを復号してから前記第２のプログラムによってその復号された２番目の処理コードを実行するようにしたことを特徴とする請求項１０から請求項１７のいずれか記載のプログラム実行方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明はプログラム構築方法及びそのプログラムの実行方法に関し、特にソフトウェアの動的な解析を阻止するための技術に関する。

【０００２】

【従来の技術】

従来、ソフトウェアに関する不正利用への対抗技術としては、例えば「逆解析や改変からソフトを守る－タンパレジスタントソフトウェア技術－」（１９９８年１月、日経エレクトロニクス通巻７０６号，ｐｐ．２０９～２２０）に記載されているように、第三者が秘密性の高いプログラムを解析し、不正利用することを防ぐために用いられている。

【０００３】

その中でも、実行中のプログラムに対してソフトウェアデバッガ等を用いることによって接触し、プログラムを一文ずつ実行させて、その挙動を調べるやりか

たは、完全な回避方法のない強力な解析手段である。

【０００４】

上記の文献に記載されている不正利用への対抗技術における動的な解析への対抗手段について説明する。その構成を図８に示す。この図８において、プログラム１０，２０は不正利用への対抗技術を適用して保護するべきプログラムであり、プログラム６０は不正利用への対抗技術のために用意した監視プログラムである。尚、プログラム１０，２０，６０は自分自身の改ざんを検出する改ざん検出コード３２をそれぞれ含んでいる。

【０００５】

これらの認証について説明する。プログラム１０とプログラム６０とはディジタル署名による通信プロトコルによって互いの改ざん検出コード３２を認証する。その際、改ざん検出コード３２が壊れたり、改ざんされている場合にはそこで処理を終了する。

【０００６】

プログラム２０とプログラム６０とはディジタル署名による通信プロトコルによって互いの改ざん検出コード３３を認証する。その際、改ざん検出コード３２が壊れたり、改ざんされている場合にはそこで処理を終了する。

【０００７】

これらの方法によって、改ざん検出コード３２が動的な解析を検出した場合には処理を中断することができる。また、２つの組と相互に認証を行うことで、通信プロトコルのメッセージを模倣してなりすます第三者のプログラムにも対応することができる。

【０００８】

【発明が解決しようとする課題】

上述した従来のソフトウェアに関する不正利用への対抗技術では、ソフトウェアデバッガによる解析でトレースのみを行っている場合、改ざん検出コードがプログラムの改ざんを検出するのみであるので、デバッガ側がプログラムのどこかを改ざんしなくてはその改ざんを検出することができないという問題がある。

【０００９】

そこで、本発明の目的は上記の問題点を解消し、ソフトウェアデバッガによる動的なソフトウェア解析を不可能にすることができるプログラム構築方法及びそのプログラムの実行方法を提供することにある。

【００１０】

【課題を解決するための手段】

本発明によるプログラム構築方法は、各々対応する処理コードを暗号化した暗号化コードをその実行時に相互に解除する第１及び第２のプログラムで構築している。

【００１１】

本発明によるプログラム実行方法は、第１及び第２のプログラムの実行時に前記第１及び第２のプログラム各々に対応する処理コードを暗号化した暗号化コードを相互に解除するようにしている。

【００１２】

すなわち、本発明のプログラム構築方法は、２つのプログラムがそれぞれ暗号化したプログラムコードを持ち、実行時に暗号化したプログラムコードを相互に解除していくことで、ソフトウェアデバッガによるプログラムの動的な解析を妨ぐことができるように構築している。

【００１３】

より具体的に、本発明のプログラム構築方法では、第１のプログラムが全体の処理動作において奇数の順序で実行される暗号化コードを含み、第２のプログラムが全体の処理動作において偶数の順序で実行される暗号化コードを含んでいる。第１及び第２のプログラムにおいては各々の暗号化コードを互いに交互に実行するように組まれている。暗号化コードはどれも第１及び第２のプログラムの両方で復号しないと元のコードに戻らないようにしておく。

【００１４】

第１及び第２のプログラムは起動されると、まず１番目の暗号化コードを復号し、第１のプログラムはその復号されたコードを実行する。次に、第１及び第２のプログラムは２番目の暗号化コードを復号し、第２のプログラムはその復号されたコードを実行する。

【００１５】

上記の処理を繰り返すことで、ソフトウェアデバッガによって第１のプログラムに動的な解析が行われた場合、暗号化コードが完全に復号されなくなり、それを実行しようとした第２のプログラムは不正なコードを実行したことによって終了し、以後の暗号化コードが完全に復号されなくなった第１のプログラムも終了することになる。

【００１６】

本発明では、ソフトウェアデバッガが起動しているひとつのプログラムしか解析することができない点と、動的な解析が通常にプログラムが実行されるよりも時間がかかるという点の２つの特徴を利用している。これによって、不正な動的解析を妨げる構造が可能となる。

【００１７】

【発明の実施の形態】

次に、本発明の実施例について図面を参照して説明する。図１は本発明の一実施例によるプログラムの構造を示す図である。図１において、プログラム１０は暗号化コード１１，１３，１５と、暗号化コード１２，１４，１６の解除コード１２ａ，１４ａ，１６ａと、復号処理コード３０とを含んでおり、プログラム２０は暗号化コード１２，１４，１６と、暗号化コード１１，１３，１５の解除コード１１ａ，１３ａ，１５ａと、復号処理コード３１とを含んでいる。

【００１８】

図２は図１の暗号化をする前のプログラムを示す図である。図２において、暗号化コード１１～１６はそれぞれ任意の処理コード１～６を暗号化したコードであり、どれも復号処理コード３０及び復号処理コード３１で復号しなければ元の処理を行う処理コード１～６に戻らない。

【００１９】

その場合、復号処理コード３０は解除コード１２ａ，１４ａ，１６ａでプログラム２０の暗号化コード１２，１４，１６の解除を行い、復号処理コード３１は解除コード１１ａ，１３ａ，１５ａでプログラム１０の暗号化コード１１，１３，１５の解除を行う。

【００２０】

図３は図１のプログラムが実行される順番を示す図である。図３においては、プログラム１０，２０で時間的にそれぞれ暗号化コード１１，１２，１３，１４，１５，１６の順番で実行するように構成されていることを示している。

【００２１】

異なるプログラムにおいて順番に実行される暗号化コードを構築するには、システムタイマ等の同期機構を利用してプログラムを構築する方法や、初期設定処理、画面描画処理、終了処理等の実行される順番が決まっている処理に、暗号化コードを組込む等の方法があるが、ソフトウェアデバッガによる解析で停止をしない仕組みでなければならない。実際にはシステムタイマ等を用いた時間によるウェイトが望ましい。

【００２２】

図４は図１のプログラムが正しく動作している場合を示す図であり、図５は図１のプログラムの動作を示すフローチャートである。これら図１と図４と図５とを参照して本発明の一実施例の全体の動作について説明する。

【００２３】

まず、プログラム１０とプログラム２０とが起動されると、復号処理コード３０による解除と、解除コード１１ａを用いた復号処理コード３１による解除とによって暗号化コード１１の暗号が解除されて処理コード１となる（図５ステップＳ１，Ｓ１１）。

【００２４】

この時、暗号解除を待つ側はウェイト機構で待機する（図５ステップＳ２）。このウェイト機構は並立して動作する２つのプログラム１０，２０において、同期をとるためのルーチンである。プログラム２０による暗号化コード１１の解除が行われるまで、予め算出した時間だけ待機する。その算出時間がすぎると、暗号化解除の有無にかかわらず、次のステップを実行する。

【００２５】

この後に、プログラム１０によって処理コード１が実行される（図５ステップＳ３）。続いて、解除コード１２ａを用いた復号処理コード３０による解除と復

号処理コード３１の解除とによって暗号化コード１２が解除されて処理コード２となる（図５ステップＳ４，Ｓ１２）。

【００２６】

この時、暗号解除を待つ側はウェイト機構で待機し（図５ステップＳ１３）、その後に、プログラム２０によって処理コード２が実行される（図５ステップＳ１４）。続いて、復号処理コード３０による解除と、解除コード１３ａを用いた復号処理コード３１による解除とによって暗号化コード１３が解除されて処理コード３となる（図５ステップＳ５，Ｓ１５）。

【００２７】

この時、暗号解除を待つ側はウェイト機構で待機し（図５ステップＳ６）、その後に、プログラム１０によって処理コード３が実行される（図５ステップＳ７）。続いて、解除コード１４ａを用いた復号処理コード３０による解除と復号処理コード３１による解除とによって暗号化コード１４が解除されて処理コード４となる（図５ステップＳ８，Ｓ１６）。

【００２８】

この時、暗号解除を待つ側はウェイト機構で待機し（図５ステップＳ１７）、その後に、プログラム２０によって処理コード４が実行される。というように、プログラム１０，２０の処理が続けられていく。

【００２９】

この手続きによって、プログラム１０とプログラム２０とのどちらかがデバッガ（図示せず）によって動的なデバッグが行われても、これらプログラム１０，２０の解析を防ぐことができる。

【００３０】

図６は図１のプログラムが動的にデバッグされている場合を示す図である。図６においては、例えばプログラム２０が実行中にデバッガ４０によって動的に解析されそうになった場合を示している。

【００３１】

この場合、プログラム１０とプログラム２０とが起動されて処理を始めると、復号処理コード３０，３１によって暗号化コード１１が解除されて処理コード１

になり、プログラム１０によって処理コード１が実行される。

【００３２】

デバッガ４０によってプログラム２０はデバッグされると、プログラム２０の実行は一時停止するか、デバッガ４０の使用者によってステップ実行され、ゆっくりとした処理速度になる。

【００３３】

復号処理コード３０によって暗号化コード１２は一部の暗号が解除されるとともに（不完全な処理コード２）、復号処理コード３０によって暗号化コード１３も一部の暗号が解除される（不完全な処理コード３）。プログラム１０によって不完全な処理コード３が実行されると、不正な処理を行ったとして、オペレーティングシステムによってプログラム１０が終了させられる。

【００３４】

復号処理コード３１によって不完全な処理コード２は処理コード２になり、プログラム２０はその処理コード２を実行する。また、復号処理コード３１によって暗号化コード１４は一部の暗号が解除され（不完全な処理コード４）、プログラム２０によってその不完全な処理コード４が実行されると、不正な処理を行ったとして、オペレーティングシステムによってプログラム２０が終了させられる。

【００３５】

上記のように、プログラム２０がデバッガ４０のデバッグによって、処理の流れが図３の流れよりも遅れると、プログラム１０は一部しか解除されない処理コードを実行し、不正な処理によって終了する。それによって、プログラム２０も一部しか解除されない処理コードを実行し、不正な処理によって終了する。デバッガ４０の解析もそこで終了することになる。

【００３６】

このように、２つのプログラム１０，２０が互いのプログラムに対して、段階的に暗号解除処理を行うことで、どちらのプログラムがデバッグされても、暗号解除処理の手順がずれて、不正な処理によって終了するという構造を持つことによって、ソフトウェアデバッガ４０による動的なソフトウェア解析を不可能にす

ることができる。

【００３７】

尚、上記の説明では２個のプログラム１０，２０について述べているが、それらのプログラムは２個ではなく、Ｎ個（Ｎは３以上の整数）でもかまわない。その場合、プログラムをＮ個に増やすと、プログラム１０とプログラム２０とが相互の暗号解除を行い、プログラムＮ－１とプログラムＮとが相互の暗号解除を行い、プログラムＮとプログラム１０とが相互の暗号解除を行うという形になる。これによって、全体的にさらに解析しにくいプログラムとなる。

【００３８】

また、図１及び図２において、復号処理コード３０，３１は処理コード１～６に含めてもかまわない。このように、任意の処理コード１～６に復号処理コード３０，３１を含めると、第三者による解析の危険をさらに減らすことができる。

【００３９】

さらに、図５に示すフローチャートにおいて、ウェイト機構は時間による待機以外にも、第三のプログラムによる同期処理でもかまわない。この場合、このプログラムはプログラム１０とプログラム２０とに暗号解除を許可するメッセージを順番に送信する。例えば、ステップＳ１１の終了を確認したら、ステップＳ２で待機しているプログラム１０にウェイト終了のメッセージを送信する。

【００４０】

この方法を使用した場合にも、ウェイト機構に、一定時間メッセージを受信しなかったら、次のステップに進む処理を入れることで、第三のプログラムへの動的な解析に対抗することができる。

【００４１】

図７は本発明の他の実施例によるプログラムの構造を示す図である。図７において、本発明の他の実施例によるプログラム１０，２０にはそれぞれ復号処理コード３０，３１を含んでおり、相互に解除する暗号化コード１１～１３はプログラム１０，２０上ではなく、別に配設した共有領域５０上にある。

【００４２】

図７に示すように、プログラム１０、２０は共有領域５０にある暗号化コード

１１～１３を解除し、それぞれの処理コードを実行していく。オペレーティングシステムの仕様で相手のプログラム上の暗号化コードの書き換えが困難な場合には、この方法によって上記と同様の効果が得られる。

【００４３】

【発明の効果】

以上説明したように本発明によれば、各々対応する処理コードを暗号化した暗号化コードをその実行時に相互に解除する第１及び第２のプログラムで構築することによって、ソフトウェアデバッガによる動的なソフトウェア解析を不可能にすることができるという効果がある。

【図面の簡単な説明】

【図１】

本発明の一実施例によるプログラムの構造を示す図である。

【図２】

図１の暗号化をする前のプログラムを示す図である。

【図３】

図１のプログラムが実行される順番を示す図である。

【図４】

図１のプログラムが正しく動作している場合を示す図である。

【図５】

図１のプログラムの動作を示すフローチャートである。

【図６】

図１のプログラムが動的にデバッグされている場合を示す図である。

【図７】

本発明の他の実施例によるプログラムの構造を示す図である。

【図８】

従来の動的解析防止方法を示す図である。

【符号の説明】

１～６　処理コード

１１～１６　暗号化コード

１１ａ，１２ａ，

１３ａ，１４ａ，

１５ａ，１６ａ　解除コード

１０，２０　プログラム

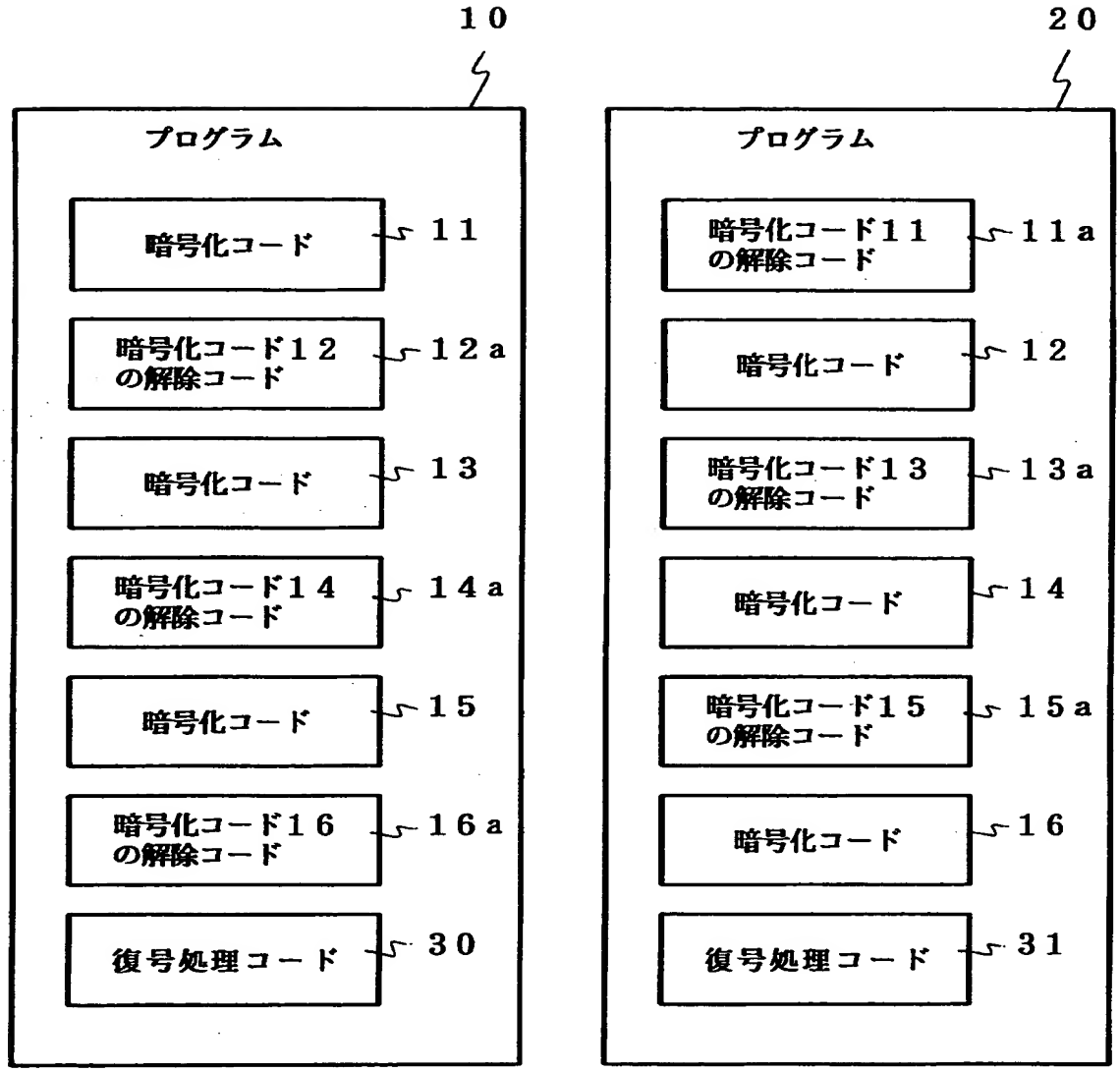
３０，３１　復号処理コード

４０　デバッガ

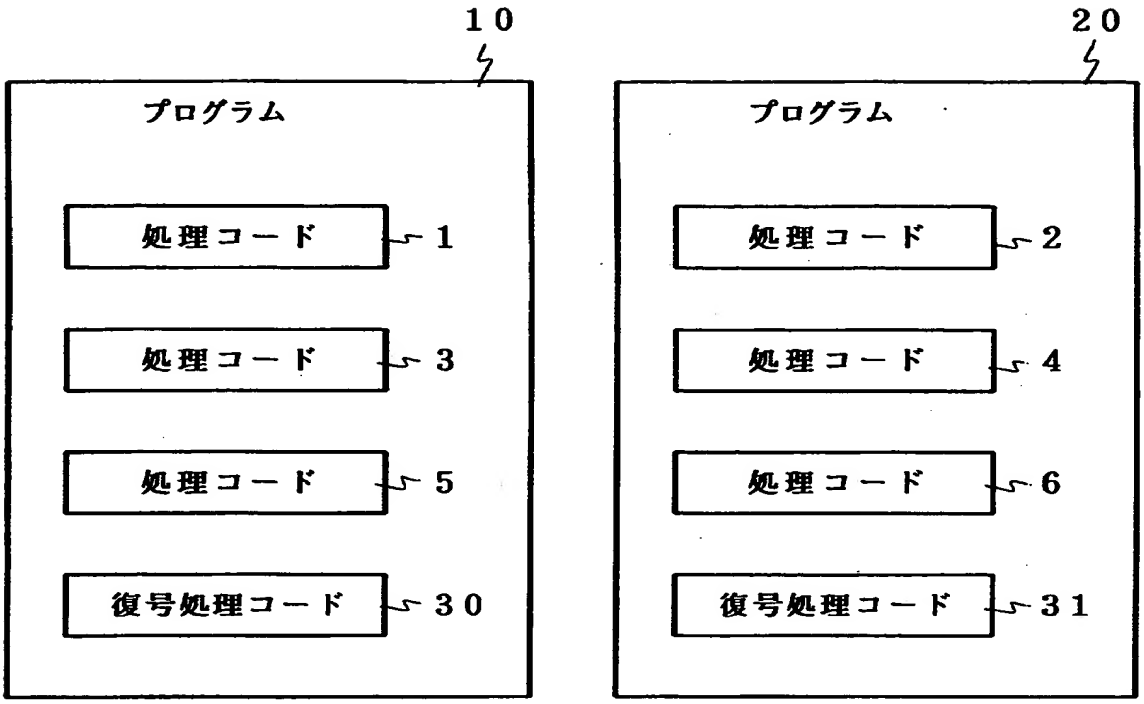
５０　共有領域

【書類名】　　図面

【図１】

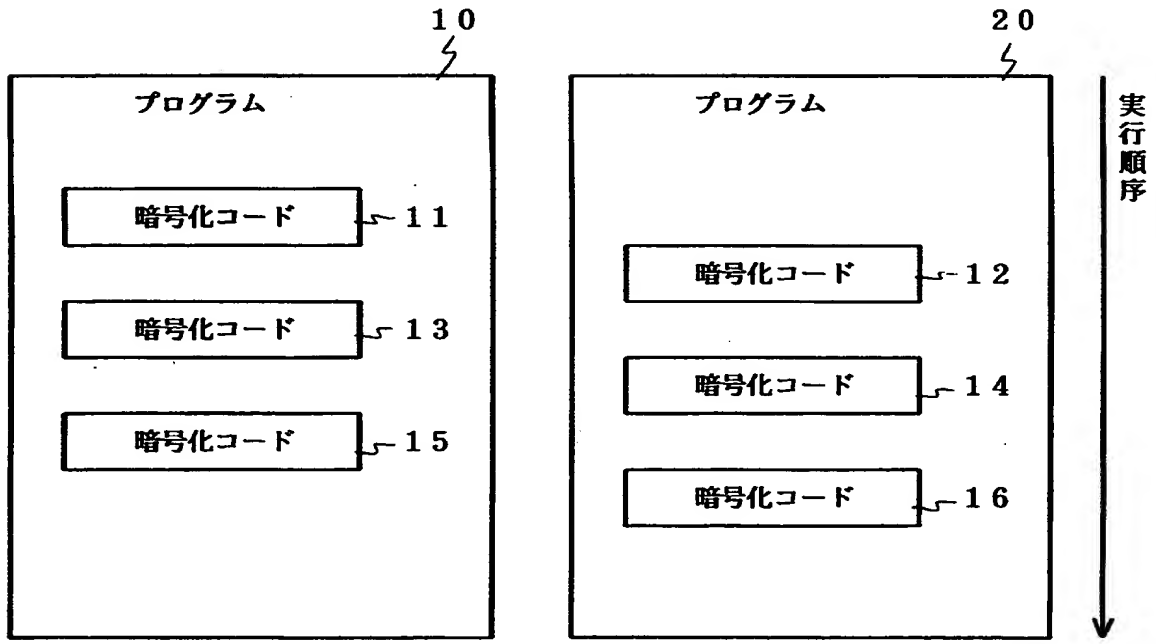

１０
プログラム
暗号化コード
１１
暗号化コード１２の解除コード
１２ａ
暗号化コード
１３
暗号化コード１４の解除コード
１４ａ
暗号化コード
１５
暗号化コード１６の解除コード
１６ａ
復号処理コード
３０
２０
プログラム
暗号化コード１１の解除コード
１１ａ
暗号化コード
１２
暗号化コード１３の解除コード
１３ａ
暗号化コード
１４
暗号化コード１５の解除コード
１５ａ
暗号化コード
１６
復号処理コード
３１

【図２】



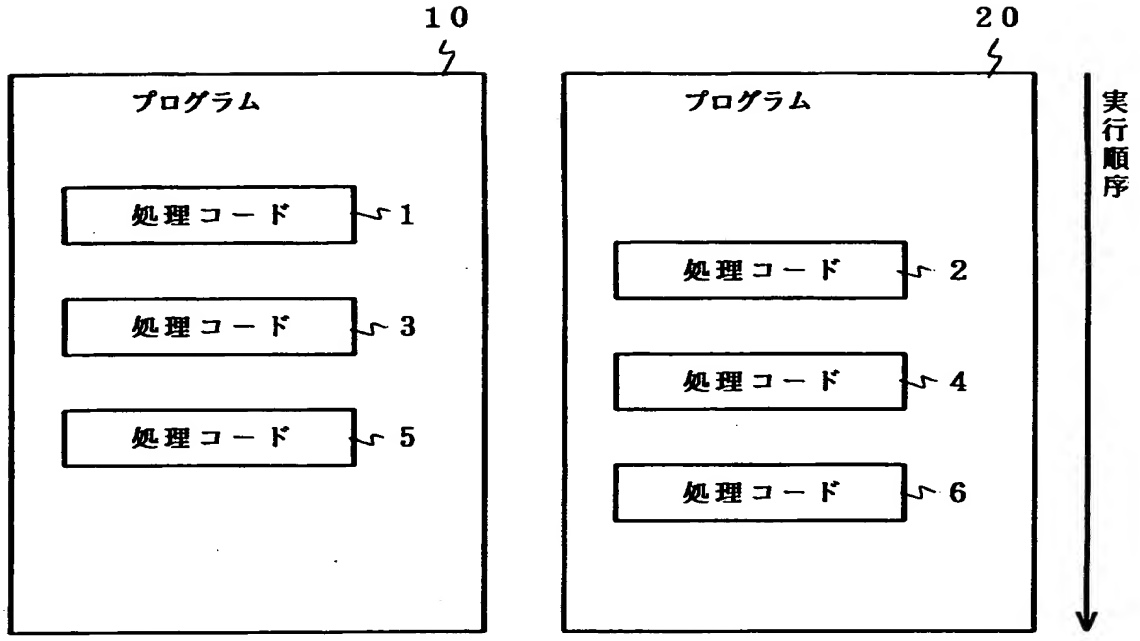
暗号化前の状態

【図３】



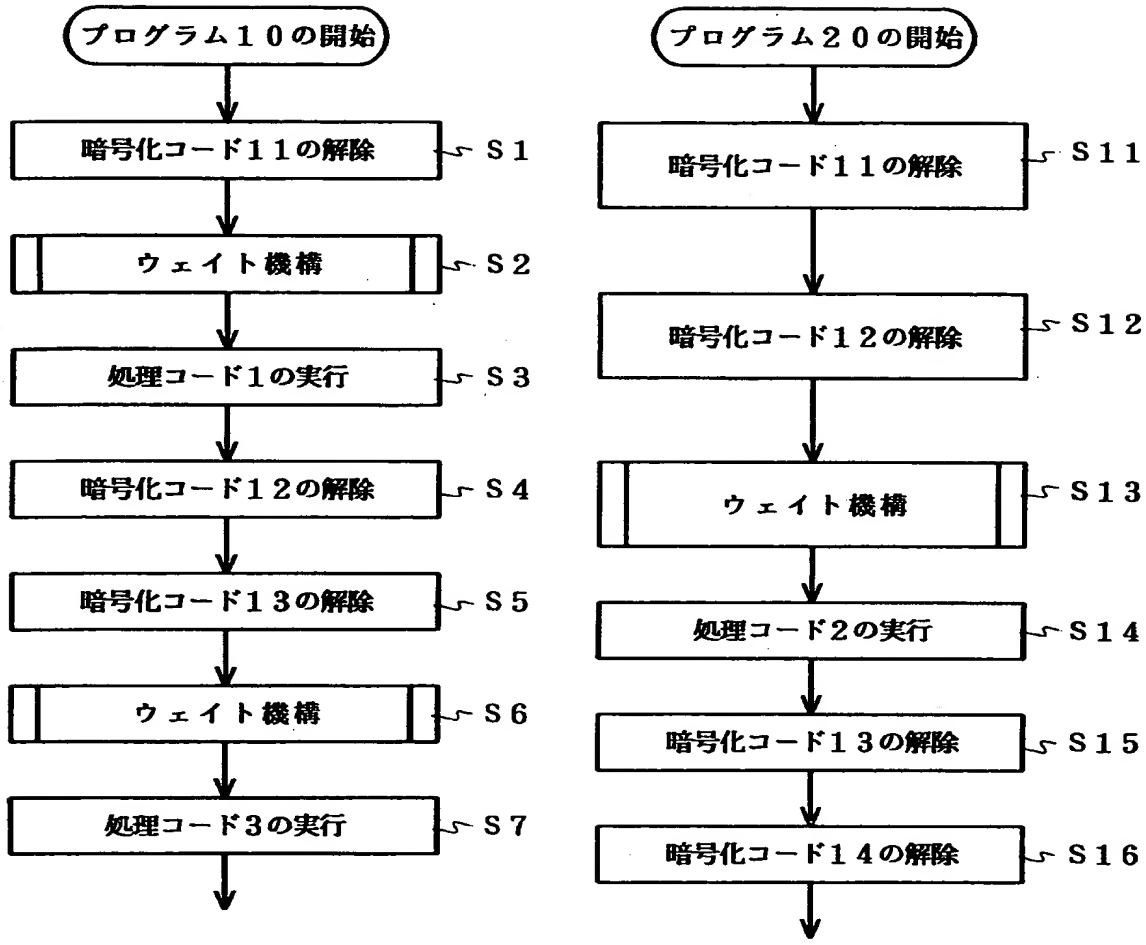
出証特２００１－３０１６６４３

【図４】

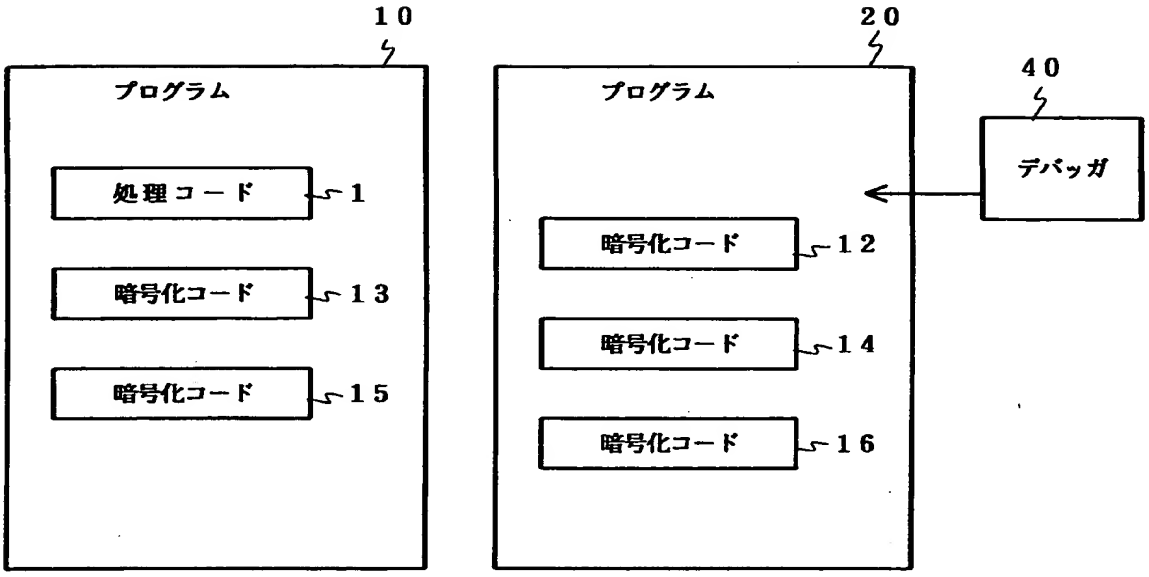


正しく動作している時

【図５】

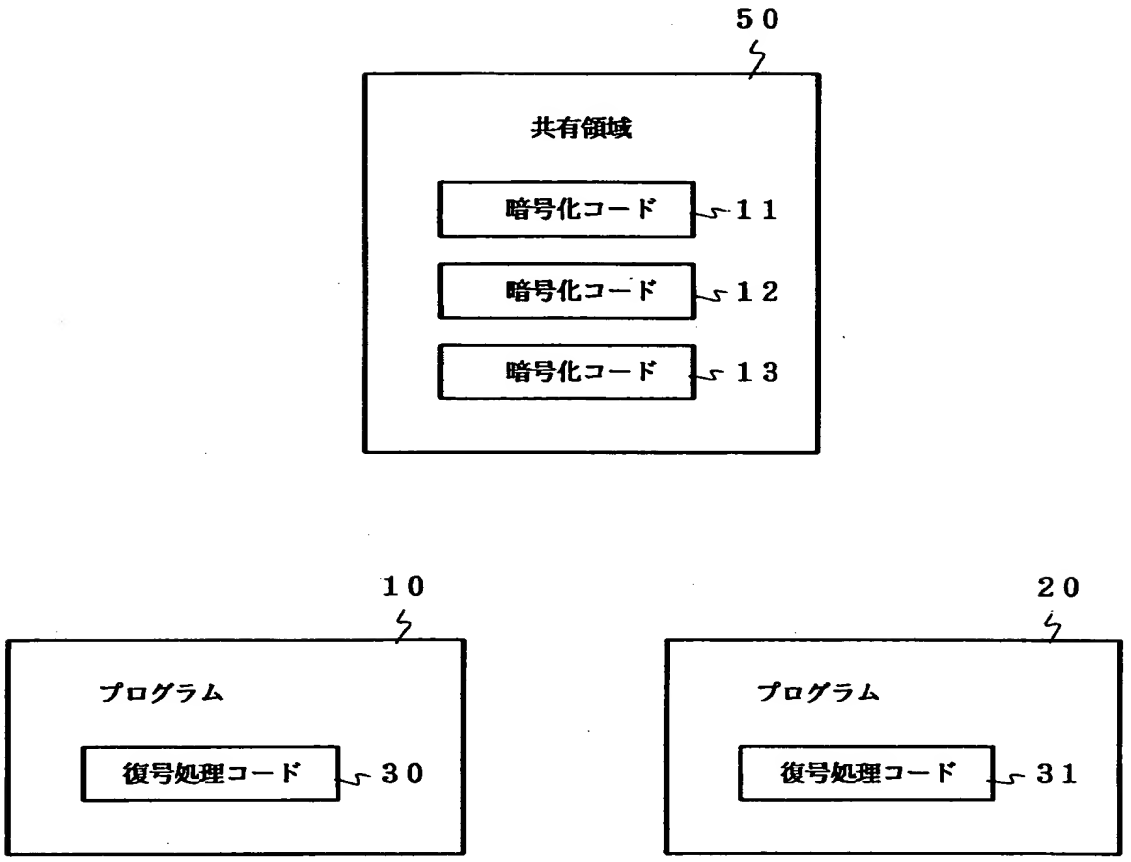

プログラム１０の開始
暗号化コード１１の解除
S１
ウェイト機構
S２
処理コード１の実行
S３
暗号化コード１２の解除
S４
暗号化コード１３の解除
S５
ウェイト機構
S６
処理コード３の実行
S７
プログラム２０の開始
暗号化コード１１の解除
S１１
暗号化コード１２の解除
S１２
ウェイト機構
S１３
処理コード２の実行
S１４
暗号化コード１３の解除
S１５
暗号化コード１４の解除
S１６

【図６】



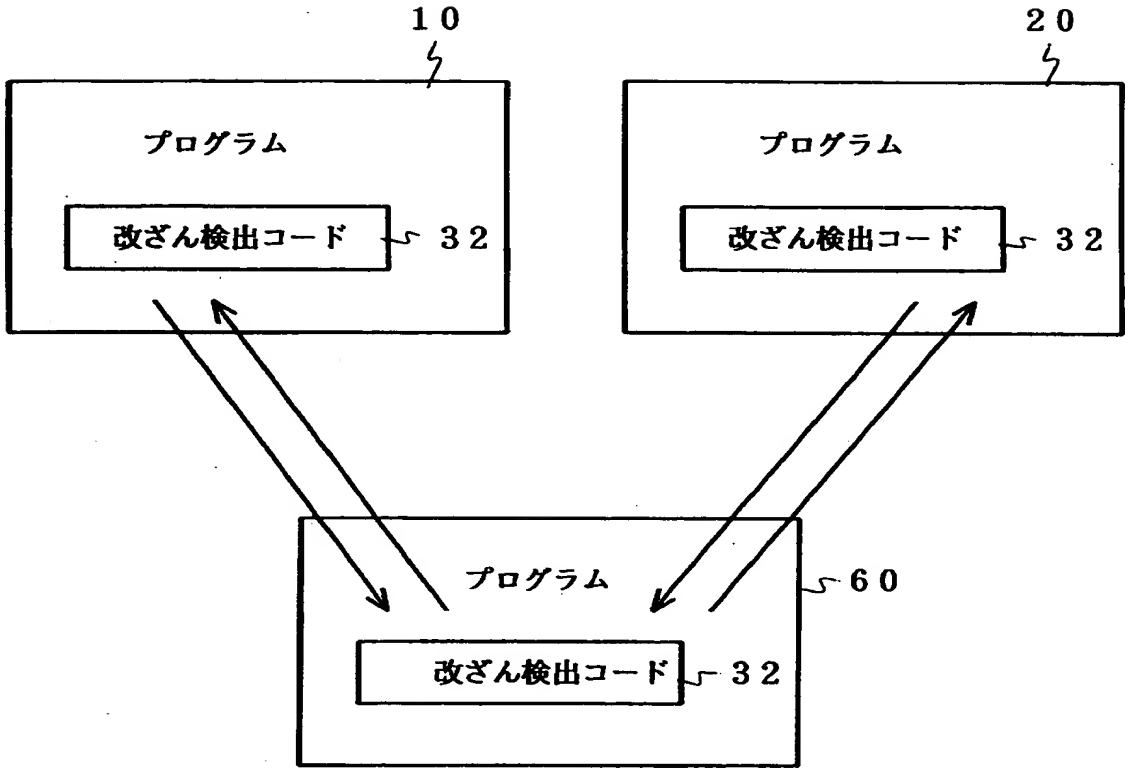
動的にデバッグされた時

【図７】



共有領域を用いる場合

【図８】


１０
２０
プログラム
改ざん検出コード
３２
プログラム
改ざん検出コード
３２
プログラム
６０
改ざん検出コード
３２

【書類名】　　要約書

【要約】

【課題】　ソフトウェアデバッガによる動的なソフトウェア解析を不可能にすることが可能なプログラム構築方法を提供する。

【解決手段】　プログラム１０は暗号化コード１１，１３，１５と、暗号化コード１２，１４，１６の解除コード１２ａ，１４ａ，１６ａと、復号処理コード３０とを含んでおり、プログラム２０は暗号化コード１２，１４，１６と、暗号化コード１１，１３，１５の解除コード１１ａ，１３ａ，１５ａと、復号処理コード３１とを含んでいる。暗号化コード１１～１６はそれぞれ任意の処理コードを暗号化したコードであり、どれも復号処理コード３０及び復号処理コード３１で復号しなければ元の処理を行う処理コードに戻らない。

【選択図】　　図１

出　願　人　履　歴　情　報

識別番号　　〔０００００４２３７〕

１．変更年月日　　１９９０年　８月２９日
　［変更理由］　　新規登録
　　住　所　　東京都港区芝五丁目７番１号
　　氏　名　　日本電気株式会社